Содержание

1 Лабораторная работа № 2. Использование криптосистемы GPG 1
1.1 Информация по стенду . . . 1
1.1.1 Задание 0. Подготовка системы, получение материалов для выполнения лабораторной . . . 1
1.2 Общая информация . . . 2
1.2.1 Задание 1. Установка . . . 2
1.3 Создание GPG-ключей . . . 2
1.3.1 Генерация ключа с расширенным набором параметров . . . 3
1.3.2 Задание 2. Генерация пары ключей . . . 4
1.4 Импорт/экспорт публичных ключей GPG . . . 5
1.4.1 Экспорт открытого ключа . . . 6
1.4.2 Задание 3. Выполнение экспорта публичного ключа . . . 6
1.4.3 Импорт открытого ключа . . . 6
1.4.4 Задание 4. Выполните импорт открытого ключа . . . 9
1.5 Шифрование файла средствами gpg . . . 9
1.5.1 Задание 5. Выполните шифрование файла средствами GPG 9
1.6 Использование GPG для подписи . . . 10
1.6.1 Создание отсоединенной подписи . . . 10
1.6.2 Создание встроенной подписи . . . 10
1.6.3 Проверка подписей . . . 11
1.6.4 Задание 6. Создание отсоединенной подписи . . . 11
1.7 Итоговое задание. Загрузка результата выполнения лабораторной работы . . . 11

1. Лабораторная работа № 2. Использование криптосистемы GPG

1.1. Информация по стенду

- Требуемые ВМ - **deb10-1**
- Вход в систему

```
sysadmin:netlab123
```

- Получение привелегий суперпользователя (пароль netlab123)

```
$ su -
```

1.1.1. Задание 0. Подготовка системы, получение материалов для выполнения лабораторной

- Установка необходимых компонент в ОС Debian

```
$ apt install git
```

Убедитесь, что git установлены.

- Получение файлов, необходимых для выполнения лабораторной работы. Выполняете в указанной ВМ, войдя в систему под УЗ **sysadmin** и находясь в его домашнем каталоге

```
$ git clone https://github.com/esorlov/infsec/
```

Если указанный репозиторий (каталог **infsec**) уже существует, перейдите в него и выполните актуализацию его содержимого следующими командами.

```
$ cd infsec
$ git pull origin master
```

- Перейдите в каталог текущей лабораторной работы

```
$ cd /home sysadmin/infsec/lab2
```

- Убедитесь, что присутствуют публичный ключ **874A9D18.asc** и файл **open-text.txt**

```
$ pwd
/home/sysadmin/infsec/lab2
$ ls
874A9D18.asc open-text.txt
```

1.2. Общая информация

- **GNU Privacy Guard (GnuPG, GPG)** - свободная ассиметричная криптосистема для шифрования информации и создания электронных цифровых подписей.

1.2.1. Задание 1. Установка

1. Установите программу gnupg (если не установлена) командой. Для этого необходимо перейти контекст безопасности суперпользователя.

```
$ apt install gnupg
```

Команда для установки gnupg в дистрибутиве Debian и порожденных.
В других дистрибутивах используйте команды соответствующих менеджеров пакетов/зависимостей.

2. *Внимание!* Далее задание выполняется от имени учетной записи **sysadmin**.

1.3. Создание GPG-ключей

- Для создания ключа используется следующая команда с базовыми или расширенными параметрами

- При создании ключа необходимо указывать полное имя и адрес электронной почты. Эта информация нужна для привязки созданного ключа к конкретному лицу.

```
$ gpg --gen-key
```

сгенерирует пару ключей GPG: открытый и секретный.

- **Открытый ключ** - общедоступен
  - используется для:
    * проверки ЭЦП
    * зашифрования сообщений.
- **Закрытый ключ** - хранится только у владельца
  - используется для:
    * установки ЭЦП
    * дешифрованию сообщений

1.3.1. Генерация ключа с расширенным набором параметров

```
$ gpg --full-generate-key
```

- **Тип ключа** - выбор используемых ассиметричных криптоалгоритмов
  - (1) RSA и RSA (по умолчанию)
  - (2) DSA и Elgamal
  - (3) DSA (только для подписи)
  - (4) RSA (только для подписи)
  - (14) Имеющийся на карте ключ

  GnuPG может создавать несколько разных типов ключей, но первичный ключ должен быть пригоден для создания подписи (signature). Поэтому в данном меню предлагается только такие варианты.

  Вариант 1 создает две пары ключей. Первичная (RSA), используемая только для подписи. Подчиненная, так же RSA используемая для шифрования. Вариант 2 похож, но другие ассиметричные криптоалгоритмы.

  Вариант 3 создает только пару DSA, которая не может использоваться для шифрования.

  Вариант 4 создаёт пару RSA, которая может использоваться только для подписи.

  Вариант 14 предполагает использование ключа на внешнем носителе.

  В любом случае, позже можно создать дополнительные подчинённые пары ключей для подписи и шифрования.

  В большинстве случаев значения параметров по умолчанию являются достаточными и для их выбора можно нажимать Enter.

Ключи RSA/RSA и DSA/ElGamal позволяет не только подписывать сообщения и файлы, но и шифровать их.

- **Размер ключа**

  Предложенный размер ключа достаточен для большинства пользователей, предоставляя высокий уровень безопасности.

- **Срок действия ключа**
  - 0 = без ограничения срока действительности (по умолчанию)
  - = срок действительности n дней
  - w = срок действительности n недель
  - m = срок действительности n месяцев
  - y = срок действительности n лет
- После подтверждения корректности всей введенной информации и запроса пароля для создаваемого ключа программа сгенерирует большое количество случайных чисел для формирования надежного ключа.
- Для увеличения энтропии генерируемой последовательности целесообразно двигать мышь, нажимать клавиши, или каким-то другим способом влиять на загруженность системы в это время. По окончании этого процесса ваш ключ сформирован и готов к использованию:

```
pub   rsa3072 2022-10-06 [SC] [   годен до: 2024-10-05]
      D34A46032057108E0B41E82EE86166BF63576CC5
uid                      User U. Users <mail@domain.org>
sub   rsa3072 2022-10-06 [E] [   годен до: 2024-10-05]
```

- **Отпечаток ключа (key fingerprint)** – это краткая “сигнатура” открытого ключа, однозначно его определяющая. Зная его, можно найти и получить открытый ключ с сервера ключей, или убедиться в подлинности имеющегося у них экземпляра ключа.

- Необязательно записывать отпечаток ключа где-либо, так как всегда можно узнать его с помощью следующей команды

```
$ gpg --fingerprint ivan@ivanov.ru
```

- Последние восемь шестнадцатеричных цифр отпечатка – это **идентификатор открытого ключа (GPG key ID)**.
- В большинстве случаев, когда вас просят указать идентификатор ключа, следует добавить к нему префикс “0x”, к примеру в выводе выше, ID ключа будет:

```
0x63576CC5
```

1.3.2. Задание 2. Генерация пары ключей

1. Запустите генератор gpg-ключа командой

```
$ gpg --gen-key
```

2. Ответьте на вопросы gpg, указав свое реальное имя и адрес электронной почты **такие же, как используются на портале ДО (Moodle)**

```
Real name: Ivan I. Ivanov
Email address: ivanov_ii@cs.spbstu.ru
```

- и подтвердите **O** ввод ранее введенных данных

```
Change (N)ame, (E)mail, or (O)kay/(Q)uit? O
```

3. Выполните ввод секретной фразы (passphrase/фраза-пароль), запомните/запишите её

Секретная фраза используется как ключ для шифрования приватного ключа. Ее потребуется вводить каждый раз, когда вы будете выполнять операции, подразумевающие необходимость доступа к вашему приватному ключу, например установка подписи или дешифрование данных.

4. После завершения будет выведен отчет о создании ключа

```
gpg: ключ 3B466A7CAFC20E98 помечен как абсолютно доверенный
gpg: создан каталог '/home/sysadmin/.gnupg/openpgp-revocs.d'
gpg: сертификат отзыва записан в '/home/sysadmin/.gnupg/openpgp-
revocs.d/86311600249C9BA8154532823B466A7CAFC20E98.rev'.
открытый и секретный ключи созданы и подписаны.

pub   rsa3072 2021-09-29 [SC] [   годен до: 2023-09-29]
      86311600249C9BA8154532823B466A7CAFC20E98
uid                      Ivan I. Ivanov <ivan@cs.spbstu.ru>
sub   rsa3072 2021-09-29 [E] [   годен до: 2023-09-29]
```

5. Выведите список доступных приватных ключей командой

```
$ gpg --list-secret-keys
```

6. Выведите список доступных ключей командой

```
$ gpg --list-keys
```

- Результаты будут похожим на ранее приведенный вывод результатов генерации gpg-ключа.

1.4. Импорт/экспорт публичных ключей GPG

- Для защищенного взаимодействия с кем-либо, необходимо обменяться ключами. Просмотреть список имеющихся открытых ключей, можно используя команду –list-keys.

```
$ gpg --list-keys
```

1.4.1. Экспорт открытого ключа

- Перед тем как послать кому-либо открытый ключ, Вы должны его экспортировать. Для этого используется команда –export. Требуется указать аргумент, идентифицирующий экспортируемый открытый ключ, например его идентификатор или адрес электронной почты.

```
$ gpg --output name.gpg --export ID
```

где **ID** - может быть идентификатором ключа или адрес электронной почты, с которым он связан

- Ключ экспортируется в двоичном формате, что бывает неудобно. GnuPG имеет опцию командной строки –armor, которая указывает на необходимость вывода в формате ASCII.

```
$ gpg --output name.gpg --armor --export ID
```

name.gpg - имя файла, куда будет записан экспортированный ключ

1.4.2. Задание 3. Выполнение экспорта публичного ключа

1. Определите идентификатор вашего, ранее сгенерированного **публичного ключа**, посмотрев список ключей GPG
2. Выполните **экспорт** этого ключа в формате **ASCII**. Результат экспорта должен быть записан в файл **pubkey.asc**. **Этот файл понадобится как результат выполнения лабораторной работы.**

1.4.3. Импорт открытого ключа

- Открытый ключ из файла может быть добавлен к связке Ваших открытых ключей при помощи команды –import.

```
$ gpg --import 874A9D18.asc
gpg: ключ DB9729FF874A9D18: импортирован открытый ключ "Egor S. Orlov (Infsec course at HSSE) <egor.or
gpg: Всего обработано: 1
gpg:                  импортировано: 1
```

Посмотреть результирующий список ключей можно командой

```
$ gpg --list-keys
/home/sysadmin/.gnupg/pubring.kbx
---------------------------------
pub   rsa3072 2021-09-29 [SC] [   годен до: 2023-09-29]
      86311600249C9BA8154532823B466A7CAFC20E98
uid         [  абсолютно ] Ivan I. Ivanov <ivan@domain.ru>
sub   rsa3072 2021-09-29 [E] [   годен до: 2023-09-29]

pub   rsa2048 2021-09-29 [SC]
      24B39968DF98CB8509378B6ADB9729FF874A9D18
```

```
uid      [ неизвестно ] Egor S. Orlov (Infsec course at HSSE) <egor.orlov@avalon.ru>
sub   rsa2048 2021-09-29 [E]
```

Ключ импортирован, но степень его доверенности обозначена как **неизвестно (unknown)**

- Достоверность импортированного ключа должна быть подтверждена. GnuPG использует гибкую и мощную модель проверки подлинности, не требующую, чтобы Вы лично проверяли достоверность каждого импортированного ключа.

  Тем не менее, достоверность некоторых ключей Вам придется проверить самому. Сначала проверяется отпечаток (fingerprint) ключа, затем ключ заверяется подписью, для подтверждения того, что он достоверен. Отпечаток ключа можно быстро просмотреть командой –fingerprint.

```
$ gpg --fingerprint egor.orlov@avalon.ru
pub   rsa2048 2021-09-29 [SC]
      24B3 9968 DF98 CB85 0937  8B6A DB97 29FF 874A 9D18
uid      [ неизвестно ] Egor S. Orlov (Infsec course at HSSE) <egor.orlov@avalon.ru>
sub   rsa2048 2021-09-29 [E]
```

- Отпечатки ключа проверяются его владельцем. Это может быть сделано при личной встрече, по телефону, или любым другим способом, гарантирующим, что Вы общаетесь с владельцем ключа. Если отпечатки полученные Вами совпадают с указанными владельцем ключа, то можете быть уверены, что обладаете достоверной копией ключа.
- После проверки отпечатков Вы можете подписать ключ. Так как проверка ключа слабое звено в криптографии с открытым ключом, то Вы должны быть совершенно уверены в ключе перед тем, как его подписывать, и всегда проверяйте отпечатки.
- Для подписи ключа необходимо перейти в режим редактирования ключа при помощи команды **–edit-key**.
  - далее, в режиме редактирования ключа, ввести команду **sign** для его подписи (заверения)
  - подтвердить выполнение подписи, ввести парольную фразу для разблокировки своего приватного ключа, в результате чего заверение будет выполнено
  - выйти командой **quit** и подтвердить сохранение изменений

```
$ gpg --edit-key egor.orlov@avalon.ru
gpg (GnuPG) 2.2.27; Copyright (C) 2021 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
```

```
pub  rsa2048/DB9729FF874A9D18
          создан: 2021-09-29     годен до: никогда     назначение: SC
     доверие: неизвестно достоверность: неизвестно
sub  rsa2048/D61AB33E33A0A0F0
          создан: 2021-09-29     годен до: никогда     назначение: E
[ неизвестно ] (1). Egor S. Orlov (Infsec course at HSSE) <egor.orlov@avalon.ru>

gpg> sign

pub  rsa2048/DB9729FF874A9D18
          создан: 2021-09-29     годен до: никогда     назначение: SC
     доверие: неизвестно достоверность: неизвестно
 Отпечаток первичного ключа: 24B3 9968 DF98 CB85 0937  8B6A DB97 29FF 874A 9D18

     Egor S. Orlov (Infsec course at HSSE) <egor.orlov@avalon.ru>

Вы уверены, что хотите подписать этот ключ
своим ключом "Ivan I. Ivanov <ivan@domain.ru>" (3B466A7CAFC20E98)?

Действительно подписать? (y/N) y

gpg> quit
Сохранить изменения? (y/N) y
```

- Интерактивный режим подписи ключа
  - **sign** - подписать
  - **check**- проверить подпись
  - **quit** - выйти
- GnuPG поинтересуется насколько тщательно Вы проверили достоверность ключа и попросит подтвердить Ваше желание подписать ключ. Подписав ключ, Вы можете просмотреть список подписей на ключе и увидеть там добавленную Вами.

```
$ gpg --list-keys
gpg: проверка таблицы доверия
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: глубина: 0   достоверных:    1   подписанных:    1   доверие: 0-
, 0q, 0n, 0m, 0f, 1u
gpg: глубина: 1   достоверных:    1   подписанных:    0   доверие: 1-
, 0q, 0n, 0m, 0f, 0u
gpg: срок следующей проверки таблицы доверия 2024-10-05
/home/sysadmin/.gnupg/pubring.kbx
---------------------------------
pub   rsa3072 2022-10-06 [SC] [   годен до: 2024-10-05]
      D34A46032057108E0B41E82EE86166BF63576CC5
uid         [  абсолютно ] User U. Users <mail@domain.org>
sub   rsa3072 2022-10-06 [E] [   годен до: 2024-10-05]
```

```
pub   rsa2048 2021-09-29 [SC]
      24B39968DF98CB8509378B6ADB9729FF874A9D18
uid      [  полное  ] Egor S. Orlov (Infsec course at HSSE) <egor.orlov@avalon.ru>
sub   rsa2048 2021-09-29 [E]
```

1.4.4. Задание 4. Выполните импорт открытого ключа

- В каталоге **infsec/lab2** с материалами лабораторной присутствует файл **874A9D18.asc** содержащий публичный ключ преподавателя с идентификатором **874A9D18**

1. Выполните импорт открытого ключа, содержащегося в указанном файле.
2. Посмотрите список публичных ключей gpg. Убедитесь, что ключ **874A9D18** импортирован, но степень его доверия обозначена как **неизвестно**
3. Выполните подписание указанного ключа своим ключем.
4. Посмотрите списко публичных ключей, убедитесь, что импортированный ключ теперь полностью доверяем (степень доверия - **полное**)

1.5. Шифрование файла средствами gpg

- Для выполнения шифрования файла **message.txt** в файл **encrypted.asc** ключом **0x12345678** используется команда

```
$ gpg -a -r 0x12345678 -o encrypted.asc -e message.txt
```

Опция **-a** является сокращенной формой записи опции –armor. При этом результирующий файл будет текстовым (ASCII), а не бинарным.

- Для выполнения расшифрования файла **encrypted.asc** ключом **0x12345678** и сохранения его в файл **decypted.txt** используется команда

```
$ gpg -r 0x12345678 -o decrypted.txt -d encrypted.gpg
```

Операция дешифрования подразумевает необходимость использования приватного ключа, поэтому у вас будет запрошена парольная фраза для его дешифрования.

1.5.1. Задание 5. Выполните шифрование файла средствами GPG

1. В материалах лабораторной работы найдите файл **open-text.txt**, откройте его в редакторе и заполните своей информацией (ФИО, группа, дата)
2. Выполните шифрование полученного файла импортированным ключем преподавателя **0x874A9D18**, записав результат в файл **close-text.asc**. **Этот файл понадобится как результат выполнения лабораторной работы.**
3. Убедитесь, что файл **close-text.asc** содержит зашифрованное средствами PGP сообщение

1.6. Использование GPG для подписи

- Подпись устанавливается с использованием **приватного ключа пользователя**
- **Отсоединенная подпись** - подпись хранится в отдельном файле
- **Присоединенная подпись (встроенная в файл подпись)** - подпись хранится в том же файле, что и подписываемые данные

1.6.1. Создание отсоединенной подписи

- Для создания **отсоединённой подписи** (подпись хранится отдельно от документа) в текстовом формате используется команда

```
$ gpg --sign --detach-sign --default-key 0xD45AB90A --armor doc.pdf
```

где 0xD45AB90A - отпечаток ключа собственной ключевой пары (приватный ключ), а doc.pdf - имя файла, который будет подписываться. Для установки подписи требуется разблокировать приватный ключ пользователя - т.е. необходимо будет указать парольную фразу.

Отсоединенная подпись в формате ASCII будет по умолчанию записана в файл имя которого образовано из имени исходного файла путем добавления к нему расширения **asc**. Имя файла с отсоединенной подписью можно переопределить опцией **-o**

- Для создания **отсоединённой подписи в двоичном формате** используется команда

```
$ gpg --sign --detach-sign --default-key 0xD45AB90A doc.pdf
```

Все параметры те же, что и ранее, но на выходе будет получен двоичный файл с расширением **sig**.

1.6.2. Создание встроенной подписи

- Для создания **встроенной в файл подписи в текстовом формате** используется команда

```
$ gpg --sign --default-key 0xD45AB90A --armor doc.pdf
```

- Для создания **встроенной в файл подписи в двоичном формате** используется команда

```
$ gpg --sign --default-key 0xD45AB90A doc.pdf
```

- При создании встроенных подписей содержимое файла - источника полностью включается внутрь, поэтому использовать данный формат не желательно из-за дублирования и значительного размера.

1.6.3. Проверка подписей

- Для проверки отсоединённой двоичное подписи из файла **doc.pdf.sig** для файла **doc.pdf** используется команда

```
$ gpg --verify doc.pdf.sig
```

Другие типы подписей проверяются аналогично

1.6.4. Задание 6. Создание отсоединенной подписи

1. Создайте отсоединенную подпись в текстовом формате для **заполненного** ранее файла **open-text.txt**

   Создание подписи подразумевает необходимость использования приватного ключа, поэтому у вас будет запрошена парольная фраза для его дешифрования

2. Полученный результат (отсоединенная подпись в текстовом формате) должен быть записан в файл **open-text.txt.asc** (поведение команды создания подписи по-умолчанию). **Этот файл понадобится как результат выполнения лабораторной работы.**

1.7. Итоговое задание. Загрузка результата выполнения лабораторной работы

- В качестве результата выполнения лабораторной работы необходимо загрузить в Moodle следующие файлы
  - файл **pubkey.asc**, полученный в задании 3 как результат экспорта Вашего публичного ключа
  - файл **close-text.asc** полученный в задании №5, в результате шифрования заполненного файла open-text.txt импортированным ключем преподавателя
  - файл **open-text.txt.asc** полученный в задании №6, в результае установки ЭЦП на файл open-text.txt